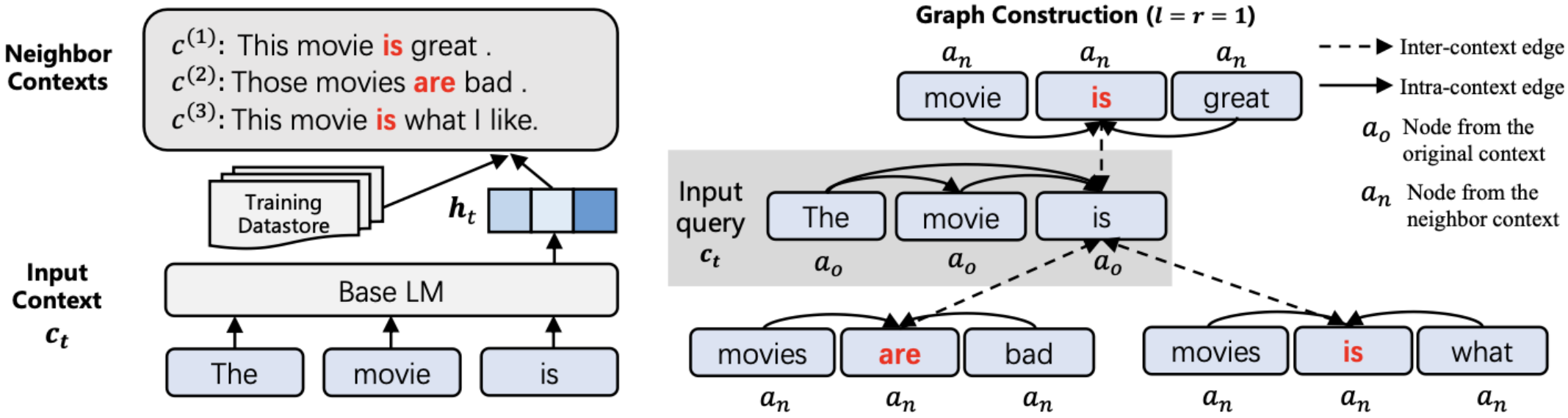
# Optimizing Graph Neural Networks for Language Modeling: Dynamic kNN, ANN, and Adaptive Graph Construction

Presented By: Firoz Shaik and Aryan Rao Neelagiri

# Authors' Application



# Research Motivation

- Inefficiencies in Static Graph Construction
- Limited Adaptability to Token Complexity
- Challenges in Scalability
- Demand for Enhanced Contextual Relevance

# Problem Statement:

The static graph construction in GNN-LM is resource-inefficient due to its uniform application of a fixed number of neighbors (kNN), leading to overuse of resources for simple tokens and insufficient context for complex ones. This approach lacks scalability and adaptability, limiting both model performance and computational efficiency, particularly in large-scale language tasks.

# Datasets

**WikiText-103:** A large word-level language modeling benchmark containing 103 million tokens, commonly used for evaluating long-term dependencies.

**One Billion Word Benchmark:** A large-scale dataset with around 768 million tokens, primarily used for language modeling with short-term dependencies.

**Enwik8:** A character-level dataset consisting of 100 million characters from English Wikipedia articles, used for evaluating character-level language modeling.

# Methods

- **Efficiency with kNN:**
  - Dynamic k: Fixed neighbor count is inefficient. Dynamic k adjusts the number of neighbors based on input complexity, optimizing accuracy and resource usage.

**Sentence 1**: "I went to the bank to deposit money."

**Sentence 2**: "The bank of the river was beautiful."

- **Scalability with ANN:**
  - ANN algorithms use heuristics to quickly identify regions of the space that are likely to contain the nearest neighbors.
- **Adaptive Graph Construction:**
  - During graph construction, not all nodes (tokens) or edges (relationships between tokens) contribute equally to the final prediction.
  - **A pruning mechanism** could be implemented to remove low-impact nodes and edges during graph construction.